SIMULATION LAB #1:
Dynamic Simulation of Jumping

Modeling and Simulation of Human Movement
BME 599

Laboratory Developers: Jeff Reinbolt, B.J. Fregly, Clay Anderson, Allison Arnold, Silvia Blemker, Darryl Thelen, Scott Delp

I. Introduction

In the study of human movement, experimental measurement is generally limited to the kinematics of the body segments, external reaction forces, and electromyographic (EMG) signals. While these data are essential for characterizing movement, important information is missing. For example, because the body is actuated by more muscles than it has degrees of freedom, we cannot uniquely solve for the muscle forces that gave rise to an observed motion. Yet, knowledge of muscle force is essential for quantifying the stresses placed on bones and also for understanding the functional roles of muscles in normal and pathological movement. Using dynamic models of the musculoskeletal system to simulate movement provides not only a means of estimating muscle forces but also a framework for investigating how the various components of the musculoskeletal system interact to produce movement.

The purpose of this lab is to introduce you to the components of a musculoskeletal model, illustrate how these components can be integrated together, and demonstrate the value of dynamic simulation. You will use an interactive dynamic simulation program to manually edit the excitation histories for the muscles of the lower extremity with the goal of making a musculoskeletal model jump as high as you can. Jumping was chosen as the activity for this lab because it has a well-defined objective (i.e., jump as high as possible) and, although still complex, its muscular coordination is relatively simple compared to walking. The musculoskeletal model you will use was used previously by Anderson and Pandy (1999) to find the "optimal" set of excitation histories for maximum-height jumping. For details concerning the model and the optimal solution, consult the attached paper by Anderson and Pandy (1999). By working through this lab, you will get a feel for the computational cost of dynamic simulation and gain some insight into the roles played by individual muscles during jumping. By examining the simulation results, you will get some exposure to what data are available from dynamic musculoskeletal models. Finally, by stepping through this lab, you will get a preview of upcoming labs that will focus in greater depth on the various components that comprise a dynamic musculoskeletal model.

II. Objectives

The purpose of this lab is to give you hands-on experience with a complex, dynamic model of the human musculoskeletal system. In the course of this lab, you will:

- Find a set of muscle excitations that produce a well-coordinated jump. The specific aim is to make the musculoskeletal model jump as high as possible without hyper-extending its joints.
- Investigate the actions of muscles when they are fired in isolation and in conjunction with other muscles.
- Compare the ground reaction forces predicted by your simulation with the forces predicted by the optimal solution found by Anderson and Pandy (1999).
- Quantify the magnitude of the articular contact forces in the hip.
- Examine the force generated by the vasti muscle group during jumping in relation to its excitation level and in relation to its maximum isometric force.

III. Background

Dynamic models of the musculoskeletal system are typically comprised of four important components: 1) the equations of motion for the body, or skeletal dynamics, 2) a representation of musculoskeletal geometry, 3) a model of muscle-tendon mechanics, and 4) a model of activation dynamics. Figure 1 illustrates how these components are combined to execute a forward dynamic simulation. Based on a set of initial states, which include the muscle activations $\vec{a}(t)$, the muscle forces $\vec{f}(t)$, the generalized speeds $\dot{\vec{q}}(t)$, and the generalized coordinates $\vec{q}(t)$, differential equations (See Eqs. (1)-(3) below) are used to compute the time rate of change of the states. Then a numerical integration is performed to compute the states at time $t + dt$. The new states are fed back and the forward dynamics process repeats, advancing the states in time until the final time of the simulation is reached. In the simulations you will conduct in this lab, a variable-step integrator is used.

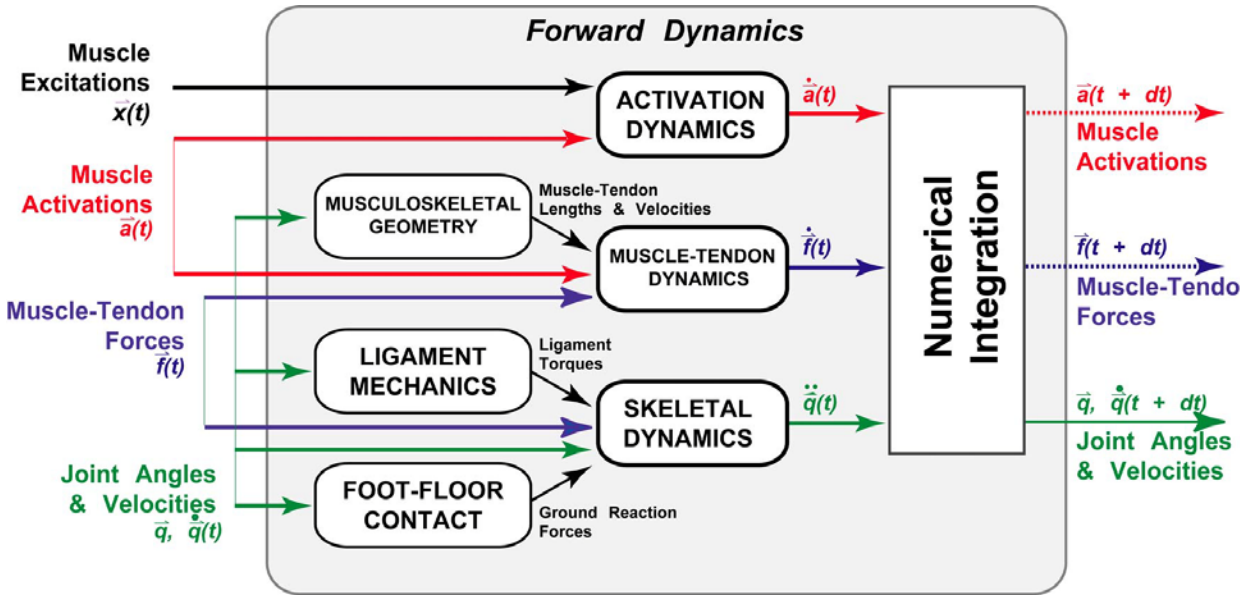

Figure 1. Schematic of a forward dynamic simulation.

Skeletal dynamics

The equations of motion for the body allow one to compute the accelerations of the body segments when forces and torques are applied to the body. The equations of motion can be expressed as follows:

$$\ddot{\vec{q}} = \overleftrightarrow{I}(\vec{q})^{-1} \cdot \left\{\vec{C}\left(\vec{q}, \dot{\vec{q}}^2\right) + \vec{G}(\vec{q}) + \overleftrightarrow{R}(\vec{q}) \cdot \vec{f}_M + \vec{E}(\vec{q}) \cdot \vec{f}_E\right\} \quad (1)$$

Eq. (1) is simply an elaboration of Newton's second law for a multi-link system, rearranged so that one can compute acceleration (i.e.,$a = m^{-1} \cdot f$). The vector of generalized coordinates, $\vec{q}$, is used to specify position and orientation of the body segments. The time derivatives of $\vec{q}$, $\dot{\vec{q}}$ & and $\ddot{\vec{q}}$, therefore represent the velocities and accelerations of the segments. Depending on how one chooses to model the body, elements of $\vec{q}$ may be translational displacements, orientations of segments with respect to the lab frame (segment angles), or orientations of segments with respect to other segments (joint angles). Implicit in one's choice of generalized coordinates are one's assumptions about the how the joints of the body function. For example, one often models the hip joint as a three degree-of-freedom ball-and-socket joint, which requires three generalized coordinates: flexion-extension (q_1), ab-adduction (q_2), and internal-external rotation (q_3). The system mass matrix, $\overleftrightarrow{I}(\vec{q})$, characterizes the inertial properties of the body (i.e., masses and moments of inertia). The remaining terms in Eq. (1) express the generalized forces or torques that act on the body. $\vec{C}\left(\vec{q}, \dot{\vec{q}}^2\right)$ represents centripetal forces that arise from the angular velocities of the segments; $\vec{G}(\vec{q})$ represents gravitational forces; $\overleftrightarrow{R}(\vec{q}) \cdot \vec{f}_M$ represents the moments applied at the joints by the muscles, and $\vec{E}(\vec{q}) \cdot \vec{f}_E$ represents external forces applied to the body such as the ground reaction force. The matrix $\overleftrightarrow{R}(\vec{q})$ is a matrix of moment arms that transform the muscle forces, $\vec{f}_M$, into joint torques. The matrix $\vec{E}(\vec{q})$ performs a similar function for the external forces, $\vec{f}_E$.

For simple models, it is possible to derive the equations of motion by hand. However, for more complex models, this is generally not feasible, and the equations of motion are generated on a computer. The jumping model used in this lab has 23 degrees of freedom (Anderson and Pandy, 1999), and the equations of motion for the jumping model were generated using OpenSim (Del et al., 2007). In subsequent labs, you will use OpenSim to generate equations of motion for models you develop (Delp et al., 1990).

Musculoskeletal geometry

Accurately representing the path of a muscle from its origin to its insertion is one of the more challenging aspects of modeling the musculoskeletal system. Sometimes a muscle can be represented as a straight-line path between its origin and insertion. Other times it is adequate to approximate the path as a series of straight-line segments which pass through a series of via points (Delp et al., 1990). When modeling muscle paths in three dimensions, it is often necessary to simulate how muscles wrap over underlying bone or musculature. Cylinders, spheres, and ellipsoids have been used as wrapping surfaces (Van der Helm et al., 1992; Garner and Pandy, 2000; Arnold et al., 2000). The jumping model used in this lab does not use wrapping surfaces; however, you should be familiar with the concept of muscle wrapping over underlying bone and musculature. In Lab 2, you will use OpenSim to specify, alter, and visualize musculoskeletal geometry for a kinematic model you develop.

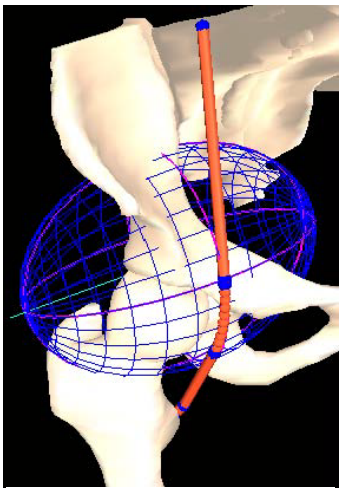

Figure 2. Path geometry of psoas.

Muscle-tendon mechanics

The force producing properties of muscle are complex and nonlinear (see McMahon (1984) for review) (Fig. 3). For simplicity, lumped-parameter dimensionless muscle models, capable of representing a range of muscles with different architectures, are most commonly used in dynamic simulation of movement (Zajac, 1989). In this lab, the jumping model is actuated by 54 musculotendinous units, each of which is represented as a Hill-type contractile element in series with tendon. The parameters used to characterize each muscle are maximum isometric force F_O^M, optimal muscle fiber length l_O^M, tendon slack length l_S^T, maximum shortening velocity V_{max}^M, and pennation angle α. For a table of the muscle-tendon parameters, consult Anderson and Pandy (1999). During a forward dynamic simulation, muscle force is treated as a state and integrated forward in time using a first-order differential equation of the form

$$\dot{f}^{MT} = \Phi(f^{MT}, l^{MT}, v^{MT}, a) \tag{2}$$

where f^{MT}, l^{MT}, and v^{MT} are the force, length, and velocity of the muscle-tendon actuator, respectively, a is the muscle activation level, and Φ is a non-linear

function (Zajac, 1989). In Lab 4, you will concentrate specifically on modeling muscle.

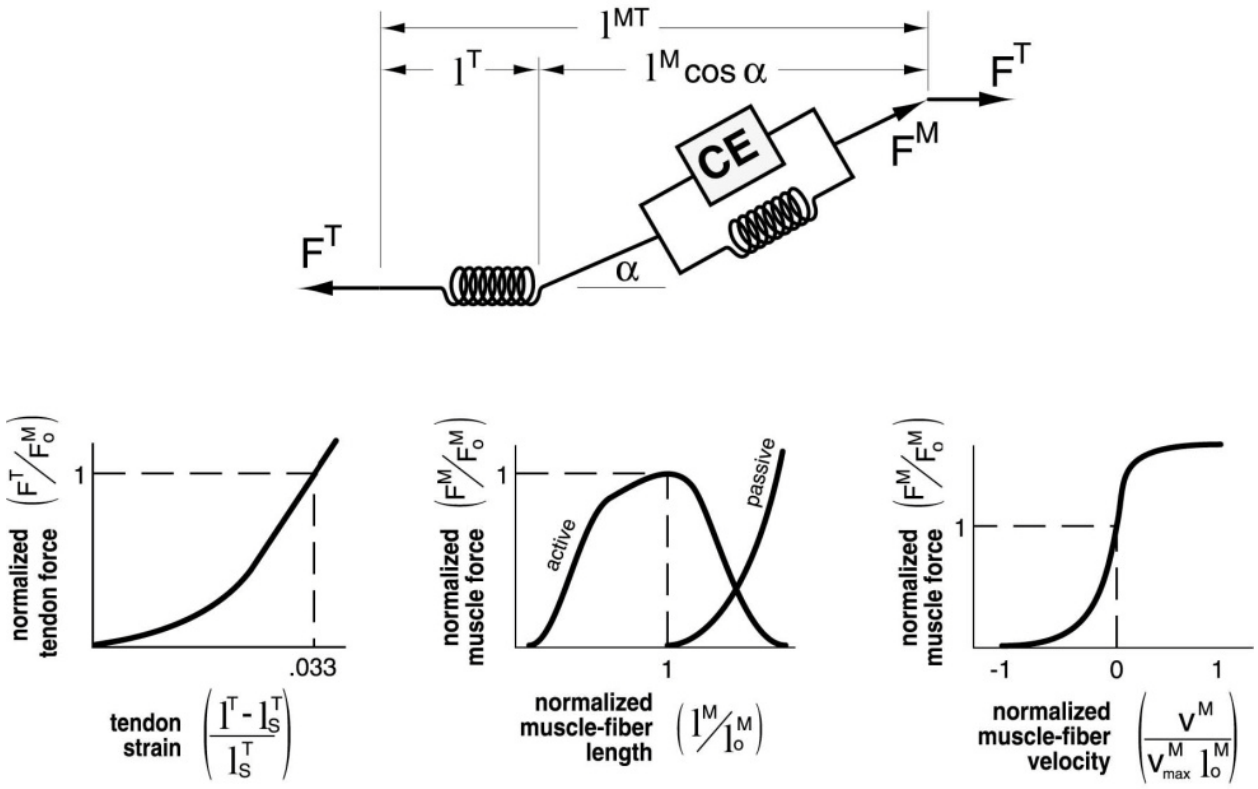

Figure 3. Dimensionless model of muscle and tendon used in our simulations. Muscle properties are represented by an active contractile element (CE) in parallel with a passive elastic element (*top*). Muscle force is dependent on muscle fiber length (*middle plot*) and velocity (*right plot*). Muscle is in series with tendon, which is represented by a nonlinear elastic element (*left plot*). Pennation angle (α) is the angle between the muscle fibers and the tendon. The forces in muscle and tendon are normalized by peak isometric muscle force (F_0^M). Muscle fiber length (l^M) and tendon length (l_S^T) are normalized by optimal fiber length (l_0^M). Tendon slack length (l_S^T) is the length at which tendons begin to transmit force when stretched. Velocities are normalized by the maximum contraction velocity of muscle (V_{max}^M). For a given muscle-tendon length (l^{MT}), velocity, and activation level, the model computes muscle force (F^M) and tendon force (F^T) .

Activation dynamics

A muscle is not capable of generating force or relaxing instantaneously. The development of force is a complex sequence of events that begins with the firing of motor units and culminates in the formation of actin-myosin cross-bridges within the myofibrils of the muscle. When the motor units of a muscle depolarize, action potentials are elicited in the fibers of the muscle and cause calcium ions to be released from the sarcoplasmic reticulum. The increase in calcium ion concentrations then initiates the cross-bridge formation between the actin and

myosin filaments (see Guyton (1986) for review). In isolated muscle twitch experiments, the delay between a motor unit action potential and the development of peak force has been observed to vary from as little as 5 milliseconds for fast ocular muscles to as much as 40 or 50 milliseconds for muscles comprised of higher percentages of slow-twitch fibers. The relaxation of muscle depends on the re-uptake of calcium ions into the sarcoplasmic reticulum. This re-uptake is a slower process than the calcium ion release, and so the time required for muscle force to fall can be considerably longer than the time for it to develop.

In the forward dynamic simulations you will conduct in this lab, activation dynamics is modeled using a first-order differential equation to relate the rate of change in activation (i.e., the concentration of calcium ions within the muscle) to excitation (i.e., the firing of motor units):

$$\dot{a} = \frac{(x^2 - xa)}{\tau_{rise}} + \frac{(x - a)}{\tau_{fall}} \quad (3)$$

where a is the activation level of a muscle, x is the excitation level of a muscle, and τ_{rise} and τ_{fall} are the rise and fall time constants for activation, respectively. In the model, activation is allowed to vary continuously between zero (no contraction) and one (full contraction). In the body, the excitation level of a muscle is a function both of the number of motor units recruited and the firing frequency of the motor units. Some models for excitation-contraction coupling distinguish these two control mechanisms (Hatze, 1976), but it is often not computationally feasible to use such models when conducting complex dynamic simulations. In the jumping model, the muscle excitation signal is assumed to represent the net effect of both motor neuron recruitment and firing frequency, and, like muscle activation, is also allowed to vary continuously between zero (no excitation) and one (full excitation). The rise and fall time constants for muscle activation are assumed to be 22 and 220 milliseconds, respectively (Zajac, 1989).

IV. Deliverables

At the completion of each lab, you will need to turn in a written report created from pre-formatted template that will be provided to you. The written reports are where you will summarize your findings and address questions posed as part of the lab. In addition, for the first lab, you will need to create plots using some of the output files from the jumper simulation.

V. Getting Started

Download the following files from the course web site:

- **Simulation_Lab1_report.docx** This Word file contains the template for the lab report.

- **Simulation_Lab1.osim** This model file contains the jumper model.
- **Simulation_Lab1_Setup_Forward.xml** This setup file contains the parameters for running the jumper simulation.
- **Simulation_Lab1_controls.xml** This controls file contains excitations for the muscles that will hold the jumping model in a static squat position.
- **Simulation_Lab1_initial_states.sto** This storage file specifies the initial values for the generalized coordinates, generalized velocities, muscle forces, and muscle activations for the jumping simulation.

Computer work

All labs for the class will be hands-on and require that you use a PC with OpenSim 2.4.0 installed.

Muscle excitation editor

In this lab, you will use OpenSim to allow you to edit the excitation signals sent to the muscles. The Excitation Editor allows you to visually inspect and edit muscle excitation patterns. To bring up the Excitation Editor, select **Edit → Excitations…** from the OpenSim main menu bar. When you press the **Load** button, you are prompted to browse for an XML file containing controls (e.g., Simulation_Lab1_controls.xml). Upon selecting the file, a filtering dialog box appears, from which you can select the controls to be displayed in the Excitation Editor for inspection and editing. Control points are selected in one of two ways, either individually or using the box select functionality. In particular, to select a control point, hold down the **ctrl key and left click** on the point. To select multiple points, hold down the **ctrl + shift keys while left clicking** on the points. You can also "box select" points by holding down the **ctrl key, and then press the left mouse button and drag the cursor to form the selection box**. If you hold down the shift key
while box selecting, you can select multiple sets of control points. Once you have selected one or more control points, you can drag them to a new location; or, you can set control points (across multiple panels) to a fixed value by typing in the new value in the text box labeled Set to. Modified excitations may be saved to the file that they came from (select **Save**) or a new controls file (select **Save As**).

See the Excitation Editor chapter in the OpenSim User Guide for additional details.

Forward Dynamic Simulation

Given the controls (e.g., muscle excitations), the Forward Dynamics Tool can drive a forward dynamic simulation. To bring up the Forward Dynamics Tool, select **Tools → Forward Dynamics…** from the OpenSim main menu bar. Like all tools, the

operations performed by the Forward Dynamics Tool apply to the current model. At the bottom of the window are four buttons. The **Settings >** button is used to load or save settings for the tool. The **Run** button starts execution. The **Close** button closes the window. The **Cancel** button closes the window and cancels all operations that have not yet been completed. When the **Settings >** button is clicked, you are presented with the choice of loading or saving settings for the tool. If you click **Load Settings…**, you will be presented with a file browser that displays all files ending with the .xml suffix. You may browse for an appropriate settings file (e.g., Simulation_Lab1_Setup_Forward.xml) and click **Open** to populate the tool with the settings in that setup file.

See the Forward Dynamics chapter in the OpenSim User Guide for additional details.

VI. Muscular Control Strategies for Maximizing Jump Height

Following are a series of tasks for you to perform and questions for you to answer. The answers to the questions constitute your written report for Lab 1. For a list of muscle abbreviations used in this lab, see Table III in Anderson and Pandy (1999).

1. Starting from the static controls (i.e., Simulation_Lab1_controls.xml), use the Excitation Editor to select all the nodes for vas_int_r and vas_int_l (collectively VAS). Then, by dragging or setting a new value for all nodes, increase the excitation of all nodes to maximum and save these controls to a new file (e.g., vas_modified_controls.xml). Next, use the Forward Dynamics Tool to start a forward integration using the newly modified controls. Let the integration complete. Briefly describe how exciting VAS affects the joint angles (*states_degrees.mot) and the ground reaction force (*forces.sto).

2. Why do a forces in VAS, which are uniarticular knee extensors, accelerate joints they do not span? Explain this dynamic coupling both physically and in terms of the inertia matrix of Eq. (1)?

3. Use the Excitation Editor to reload the static equilibrium controls (i.e., Simulation_Lab1_controls.xml). Repeat exercise 1 above for soleus_r and soleus_l (SOL).

4. Use the Excitation Editor to reload the static equilibrium controls (i.e., Simulation_Lab1_controls.xml). Repeat exercise 1 above for med_gas_r and med_gas_l (GAS).

5. Use the Excitation Editor to reload the static equilibrium controls (i.e., Simulation_Lab1_controls.xml). Repeat exercise 1 above for glut_max2_r and glut_max2_l (GMAXM) and glut_max1_r and glut_max1_l (GMAXL) together.

6. Use the Excitation Editor to reload the static equilibrium controls (i.e., Simulation_Lab1_controls.xml). Repeat exercise 1 above for bifemlh_r, bifemlb_l, bifemsh_r, and bifemsh_l (HAMS).

7. Use the Excitation Editor to reload the static equilibrium controls (i.e., Simulation_Lab1_controls.xml). Repeat exercise 1 above for add_mag2_r and add_mag2_l (ADM).

8. By manually editing the muscle excitation histories, find a set of muscle excitation patterns that produce a well-coordinated jump. Try to maximize overall performance which is jump height minus ligament force penalties. Jump height for the model is defined as the height reached by the center of mass above the model's standing height (0.9633) in meters. The ligament penalty is the integral of ligament joint torques over the duration of the simulation multiplied by a constant (see Anderson and Pandy, 1999 for details). The performance numbers can be computed on your own or you can use the included MATLAB script (i.e., jumpPerformance.m). Record your best performance numbers in your written report, and save the corresponding set of controls to a file (e.g., Simulation_Lab1_controls_best.xml).

   ligament penalty =
   jump height =
   overall performance =

9. If you are able to get the model to jump anywhere near the jump height predicted by the optimal solution (i.e., over 0.37 meters), you should be congratulated. It is not easy to do. In the more likely event that your solution was not as high as the optimal solution, explain why.

10. Choose a muscle to eliminate, and, as you did in exercise 8, make the model jump as high as possible without using this muscle. Save the controls corresponding to your best performance to a file (e.g., Simulation_Lab1_controls_second_best.xml) and again record your best performance numbers in your written report.

    ligament penalty =
    jump height =
    overall performance =

    What is the performance difference between this jump and your best jump when you could use all the muscles? What would you infer is the function of this muscle during jumping?

VII. Analyses

You will now take a brief look at some of the data which is available from your simulation.

The following files should be generated:

- **Jumper_ForceReporter_forces.sto** Time history of the muscle forces, ground reaction forces (individual contact point forces), and ligament forces
- **Jumper_JointReaction_ReactionLoads.sto** Time history of the articular contact forces at the joints
- **Jumper_controls.sto** Time history of muscle excitations
- **Jumper_states.sto** Time history of muscle activations along with other states (muscle fiber lengths and joint positions and velocities)

1. Plot the vertical ground reaction force (sum of 5 contact point forces per foot) normalized by body weight predicted by your solution and by the optimal solution. The mass of the model is 75.1658 kg, and the acceleration due to gravity is assumed to be 9.80665 m/s2. Include this plot in your report. The vertical ground reaction force (Fy) for the optimal solution is in the file **Simulation_Lab1_optimal_ground_reactions.xlsx**. Does your ground reaction force have a higher or lower peak? Is the time to lift-off longer or shorter?

2. Plot the resultant articular contact force at the hip normalized by body weight. Include this plot in your written report. Why are the hip contact forces so large?

3. On a plot, superimpose the excitation levels, activation history, and normalized force history predicted by your solution for VAS. To normalize the force predicted by VAS, divide by the force by the maximum isometric strength of VAS (6865 N, see Anderson and Pandy, 1999). Include this plot in your written report. Given what you know about muscle mechanics, explain why the force generated by VAS was less than its isometric strength?

VIII. References

1. Anderson FC and Pandy MG (1999). A dynamic optimization solution for jumping in three dimensions. Computer Methods in Biomechanics and Biomedical Engineering, 2, 201-231.
2. Arnold AS, Salinas S, Asakawa DJ, Delp SL (2000). Accuracy of muscle moment arms estimated from MRI-based musculoskeletal models of the lower extremity. Computer Aided Surgery, 5, 108-119.
3. Atkinson LV, Harley PJ, Hudson JD (1989). Numerical methods with FORTRAN 77. Addison-Wesley Publishing Company, Menlo Park.
4. Delp SL, Loan JP, Hoy MG, Zajac FE, Topp ET, Rosen JM (1990). An interactive graphics-based model of the lower extremity to study orthopaedic surgical procedures. IEEE Transactions in Biomedical Engineering, BME-37, 757-767.
5. Garner BA, Pandy MG (2000). The obstacle-set method for representing muscle paths in musculoskeletal models. Computer Methods in Biomechanics and Biomedical Engineering, 3, 1-30.

6. Guyton AC (1986). Textbook of medical physiology, Seventh Edition. W. B. Saunders Company, Philadelphia.
7. Hatze H (1976). The complete optimization of human motion. Mathematical Biosciences, 28, 99¬
8. 135. McMahon TA (1984). Muscles, Reflexes, and Locomotion. Princeton University Press,
9. Princeton, New Jersey.
10. Symbolic Dynamics, Inc. (1996). SD/FAST User's Manual, Version B.2. Mountain View, CA.
11. Van der Helm FCT, Veeger HEJ, Pronk GM, Van der Woude LHV, Rozendal RH (1992). Geometry parameters for musculoskeletal modeling of the shoulder system. Journal of Biomechanics, 2, 129-144.
12. Zajac FE (1989). Muscle and tendon: properties, models, scaling, and application to biomechanics and motor control. CRC Critical Reviews in Biomedical Engineering (Edited by Bourne JR), 17, 359-411. CRC Press, Boca Raton.